INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Introducing numerical bounds to improve event-based neural network simulation

Bruno Cessac*† , Olivier Rochel†, Thierry. Viéville †

N° 6924

Mai 2009

Thème BIO

ISSN 0249-6399 ISRN INRIA/RR--6924--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
SOPHIA ANTIPOLIS - MÉDITERRANÉE

Introducing numerical bounds to improve event-based neural network simulation

Bruno Cessac*† *, † Olivier Rochel†, Thierry. Viéville †

Thème BIO — Systèmes biologiques
Équipes-Projets Cortex

Rapport de recherche n° 6924 — Mai 2009 — 31 pages

Abstract: Although the spike-trains in neural networks are mainly constrained by the neural dynamics itself, global temporal constraints (refractoriness, time precision, propagation delays, ..) are also to be taken into account. These constraints are revisited in this paper in order to use them in event-based simulation paradigms.

We first review these constraints, and discuss their consequences at the simulation level, showing how event-based simulation of time-constrained networks can be simplified in this context: the underlying data-structures are strongly simplified, while event-based and clock-based mechanisms can be easily mixed. These ideas are applied to punctual conductance-based generalized integrate-and-fire neural networks simulation, while spike-response model (SRM) simulations are also revisited within this framework.

As an outcome, a fast minimal complementary alternative with respect to existing simulation event-based methods, with the possibility to simulate interesting neuron models is implemented and experimented.

Key-words: Spiking neural network, Neural code, Simulation.

* Laboratoire J. A. Dieudonné, U.M.R. C.N.R.S. N 6621, Université de Nice Sophia-Antipolis,France
† Cortex & Neuromathcomp EPI, INRIA, 2004 Route des Lucioles, 06902 Sophia-Antipolis, France.

Centre de recherche INRIA Sophia Antipolis – Méditerranée
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Utilisation de bornes numériques pour améliorer la simulation évennementielle de réseaux de neurones

Résumé : Bien que les trains de spikes des réseaux de neurones soient principalement contraints par la dynamique neuronale elle-même, des contraintes temporelles globales (période réfractaire, précision temporelle limitée, délais de propagation, etc...) sont aussi à prendre en compte. Ces contraintes sont revisitées dans ce papier de façon à être utilisées lors de simulations évènementielles.

Nous commencons par revoir ces contraintes et discutons leurs conséquences au niveau de la simulation, montrant comment la simulation évènementielle de réseaux contraints temporellement peut être simplifiée dans ce contexte: Ces idées sont appliquées aux modèles de ponctuels généralisés de neurones intègre-et-tire basé sur des conductances synaptiques, tandis que les modèles de type SRM sont aussi pris en compte dans ce cadre.

En terme de résultat, une alternative minimale et rapide de simulation est mise à disposition, avec la possibilité de l'utiliser dans le cas où les performances de simulation sont critiques.

Mots-clés : Réseaux de neurones évènementiels, Code neuronal, Simulation.

1 Introduction

Let us consider the simulation of large-scale networks of neurons, in a context where the spiking nature of neurons activity is made explicit [16], either from a biological point of view or for computer simulation. From the detailed Hodgkin-Huxley model [18], (still considered as the reference but unfortunately intractable when considering neural maps), back to the simplest integrated and fire (IF) model, a large family of continuous-time models have been produced, often compared with respect to their (i) biological plausibility and their (ii) simulation efficiency.

As far as this contribution is concerned, we consider a weaker notion of biological plausibility: a simulation is biologically plausible if it verifies an explicit set of constraints observed in biology. More precisely, we are going to consider a few global time constraints and develop their consequences at the simulation level. It appears here that these biological temporal limits are very precious quantitative elements, allowing us on one hand to bound and estimate the coding capability of such systems and, on the other hand, to improve simulations.

Simulation efficiency of neural network simulators

Simulation efficiency is a twofold issue of precision and performances. See [4] for a recent review about both event-based and clock-based simulation methods.

Regarding precision, event-based simulations, in which firing times are not regularly discretized but calculated event by event at the machine precision level, provides (in principle) an *unbiased* solution. On the reverse, it has been shown that a regular clock-based discretization of continuous neural systems may introduce systematic errors, with drastic consequences at the numerical level, even when considering very small sampling times [39].

Furthermore, the computational cost is in theory an order of magnitude better using event-based sampling methods [5], although this may be not always the case in practice [29], as further discussed in this paper.

However, using event-based simulation methods is quite tiresome: Models can be simulated if and only if the next spike-time can be explicitly computed in reasonable time. This is the case only for a subset of existing neuron models so that not all models can be used. An event-based simulation kernel is more complicated to use than a clock-based. Existing simulators are essentially clock-based. Some of them integrate event-based as a marginal tool or in mixtures with clock-based methods [4]. According to this collective review, only the fully supported, scientifically validated, pure event-based simulators is MVASpike [36], the NEURON software proposing a well-defined event-based mechanism [17], while several other implementations (e.g.: DAMNED [31], MONSTER [38]) exists but are not publicly supported.

In other words, event-based simulation methods may save precision and computation time, but not the scientist time.

The goal of this paper is to propose solutions to overcome these difficulties, in order to have an easy to use unbiased simulation method.

Considering integrate and fire models.

At the present state of the art, considering adaptive, bi-dimensional, non-linear, integrate-and-fire model with conductance based synaptic interaction (as e.g. in [14, 3, 39]),

called "punctual conductance based generalized integrate and fire models" (gIF), presents several advantages:

- They seem to provide an effective description of the neuronal activity allowing to reproduce several important neuronal regimes [20], with a good adequacy with respect to biological data, especially in high-conductance states, typical of cortical in-vivo activity [13].

- They provide nevertheless a simplification of Hodgkin-Huxley models, useful both for a mathematical analysis and numerical simulations [16, 19].

In addition, though these models have mainly considered one neuron dynamics, they are easy to extend to network structure, including synaptic plasticity [27, 32]. See, e.g. [33] for further elements in the context of experimental frameworks and [7, 8] for a review.

After a spike, it is assumed in such integrate and fired models that an *instantaneous* reset of the membrane potential occurs. This is the case for all models except the Spike Response Model of [16]. From the information theoretic point of view, it is a temptation to relate this spurious property to the *erroneous* fact that the neuronal network information is not bounded. In the biological reality, time synchronization is indeed not instantaneous (action potential time-course, synaptic delays, refractoriness, ..). More than that, these biological temporal limits are very precious quantitative elements, allowing one to bound and estimate the coding capability of the system.

Taking time-constraints into account

The output of a spiking neural network is a set of events, defined by their occurrence times, up to some precision:

$$\mathcal{F} = \{\cdots t_i^n \cdots\}, t_i^1 < t_i^2 < \cdots < t_i^n < \cdots, \forall i, n$$

where t_i^n corresponds to the nth spike time of the neuron of index i, with related inter-spike intervals $d_i^n = t_i^n - t_i^{n-1}$.

In computational or biological contexts, not all sequences $\mathcal{F}$ correspond to spike trains since they are constrained by the neural dynamic, while temporal constraints are also to be taken into account [10]. This is the key point here: Spike-times are
[C1] bounded by a refractory period r,
[C2] defined up to some absolute precision δt, while
[C3] there is always a minimal delay dt for one spike to propagate to another unit, and there might be (depending on the model assumptions) a
[C4] maximal inter-spike interval D such that either the neuron fires within a time delay $< D$ or remains quiescent forever). For biological neurons, orders of magnitude are typically, in milliseconds:

r	δt	dt	D
1	0.1	$10^{-[1,2]}$	$10^{[3,4]}$

The derivations of these numerical values are reviewed elsewhere [10]. This has several consequence. On one hand, this allows us ti derive an upper bound for the amount of information:

$$N \frac{T}{r} \log_2 \left(\frac{T}{\delta t}\right) \text{ bits during } T \text{ seconds,}$$

while taking the numerical values into account it means for large T, a straight-forward numerical derivation leads to about $1Kbits/neuron$.

On the other hand [9, 11], it appears that for generalized integrate and fire neuron models with conductance synapses and constant external currents, the raster plot is generically periodic, with arbitrary large periods, while there is a one-to-one correspondence between orbits and rasters. This last fact, and the fact that more general

models such as Hodgkin-Huxley neuron assemblies can be simulated during a bounded period of time [10] provides theoretical justifications for the present work.

What is the paper about

We develop here the consequences of the reviewed time constraints at the simulation level. Section 2 shows how event-based simulation of time-constrained networks can be impacted and somehow improved in this context. Section 3 considers punctual conductance based generalized integrate and fire neural network simulation, while section 4 revisits spike-response model simulation within this framework. These mechanisms are experimented in section 5, where the computer implementation choices are discussed.

Since the content of this paper requires the integration of data from the literature reused here, we have collected these elements in the appendix in order to ease the main text reading, while maintaining the self-completeness of the contribution.

2 Event-based simulation of time-constrained networks.

Clock-based and event-based simulations of neural networks make already use at different level of the global time-constraints reviewed here. See e.g. [4] for an introduction and a large review and [36, 5, 38, 30] for simulations with event-based or hybrid mechanisms. However, it appears that existing event-based simulation mechanisms gain at being revisited.

In event-based simulation the exact simulation of networks of units (e.g. neurons) and firing events (e.g. spikes) fits in the discrete event system framework [36] and is defined at the neural unit level by :

-1- the calculation of the next event-time (spike firing),
-2- the update of the unit when a new event occurs.

At the network level, the following two-stage mechanism completely implements the simulation:

-a- retrieve the next event-time and the related unit,
-b- require the update of the state of this unit, inform efferent units that this unit has emitted an event, and update the related event-times,

repeating -a- and -b- when ever events occur or when some bound is reached.

This mechanism may also take external events into account (i.e., not produced by the network units, but by external mechanisms).

Such a strategy is thus based on two key features:

- The calculation of the next event-time, conditioned to the present state and to the fact that, by definition, no event is received in the meanwhile for each unit;
- The "future" event-time list, often named "priority queue", where times are sorted and in which event-times are retrieved and updated.

The goal of this section is to revisit these two features considering [C3] and [C4].

Let us consider in this section a network with N units, an average of C connections per units, with an average number $M \leq N$ of firing units.

2.1 Event-time queue for time-constrained networks

Although, in the general case, spike-times must be sorted among pending events, yielding a $O(\log(M))$ complexity for each insertion, there exists data structure allowing to perform retrieve/update operations in constant "$O(1)$" time. Several efficient data structures and algorithms have been proposed to handle such event scheduling task. They are usually based on heap-like structures [36] or sets of buffers associated with some time intervals (such as the calendar queue in [5]). Ring buffers with fixed time step are used in [29].

The basic idea of these structures is to introduce buckets containing event-times in a given time interval. Indexing these buckets allows one to access to the related times without considering what is outside the given time interval. However, depending on the fixed or adaptive bucket time intervals, bucket indexing mechanisms and times list managements inside a bucket, retrieve/update performances can highly vary.

Let us now consider [C3] and assume that the bucket time-interval is lower than dt, the propagation delay, lower than the refractory period r and the spike time precision δt. If an event in this bucket occurs, there is at least a dt delay before influencing any other event. Since other events in this bucket occurs in a dt interval they are going to occur before being influenced by another event. As a consequence, they do not influence each others. They thus can be treated independently. This means that, in such a bucket, events can be taken into account in any order (assuming that for a given neuron the synaptic effect of incoming spikes can be treated in any order within a dt window, since they are considered as synchronous at this time-scale).

It thus appears a simple efficient solution to consider a "time histogram" with dt large buckets, as used in [29] under the name of "ring buffer". This optimization is also available in [36] as an option, while [5] uses a standard calendar queue, thus more general, but a-priory less tuned to such simulation. Several simulation methods take into account [C3] (e.g. [22, 12]).

The drawback of this idea could be that the buffer size might be huge. Let us now consider [C4], i.e. the fact that relative event-time are either infinite (thus not stored in the time queue) or bounded. In this case with $D = 10^{[3,4]}ms$ (considering fire rate down to $0.1Hz$) and $10^{-[1,2]}$ (considering gap junctions) the buffer size $S = D/dt = 10^{5-6}$, which is easily affordable with computer memories. In other words, *thanks to biological order of magnitudes reviewed previously, the histogram mechanism appears to be feasible*.

If [C4] does not hold, the data-structure can be easily adapted using a two scale mechanism. A value of D, such that almost all relative event-time are lower than D is to be chosen. Almost all event-times are stored in the initial data structure, whereas larger event-times are stored in a transient calendar queue before being reintroduced in the initial data structure. This add-on allows one to easily get rid of [C4] if needed, and still makes profit of the initial data structure for almost all events. In other words, this idea corresponds to considering a sliding window of width D to manage efficiently events in a near future. This is not implemented here, since models considered in the sequel verify [C4].

The fact that we use such a time-histogram and treat the events in a bucket in any order allows us to drastically simplify and speed-up the simulation.

Considering the software implementation evaluated in the experimental section, we have observed the following. Event retrieval requires less than 5 machine operations and event update less than 10, including the on-line detection of [C3] or [C4] violation.

The simulation kernel[1] is minimal (a 10Kb C++ source code), using a $\mathcal{O}(D/dt + N)$ buffer size and about $\mathcal{O}(1 + C + \epsilon/dt) \simeq 10 - 50$ operations/spike, thus we a small overhead $\epsilon \ll 1$, corresponding to the histogram scan. In other words, the mechanism is "$O(1)$" as for others simulation methods. Moreover, the time constant is minimal in this case. Here, we save computation time, paying the price in memory.

Remarks

The fact that we use such a time-histogram does not mean that we have discretized the event-times. The approximation only concerns the way how events are processed. Each event time is stored and retrieved with its full numerical precision. Although [C2] limits the validity of this numerical value, it is indeed important to avoid any additional cumulative round-off. This is crucial in particular to avoid artificial synchrony [36, 29].

Using [C3] is not only a "trick". It changes the kind of network dynamics that can be simulated. For instance, consider a very standard integrate and fire neuron model. It can not be simulated in such a network, since it can instantaneously fire after receiving a spike, whereas in this framework adding an additional delay is required. Furthermore, avalanche phenomena (the fact that neurons instantaneously fire after receiving a spike, instantaneously driven other neurons and so on..) cannot occur. A step further, temporal paradoxes (the fact, e.g., that a inhibitory neuron instantaneously fires inhibiting itself thus . . is not supposed to fire) cannot occur and have not to be taken into account. When considering the simulation of biological systems, [C3] indeed holds.

Only sequential implementation is discussed here. The present data structure is intrinsically sequential. In parallel implementations, a central time-histogram can distribute the unit next-time and state update calculation on several processors, with the drawback that the slower calculation limits the performance. Another idea is to consider several time-histograms on different processors and synchronize them. See [31] and [30] for developments of these ideas.

The fact that we use such a tiny simulation kernel has several practical advantages. e.g. to use spiking network mechanisms in embedded systems, etc.. However, it is clear that this is not "yet another" simulator because a complete simulator requires much more that an event queue [4]. On the contrary, the implementation has been designed to be used as a plug-in in existing simulators, mainly MVASpike [36].

2.2 From next event-time to lower-bound estimation

Let us now consider the following modification in the event-based simulation paradigm. Each unit provides:

> -1'- the calculation of *either* the next event-time,
> *or* the calculation of a lower-bound of the next event-time,
> -2- the update of the neural unit when an internal or external event occurs,
> *with* the indication whether the previously provided next event-time or lower-bound is still valid.

At the network level the mechanism's loop is now:

[1] Source code available at `http://enas.gforge.inria.fr`.

-a’- retrieve *either* the next event-time and proceed to -b’- *or* a lower-bound and proceed to -c-

-b’- require the update of the state of this unit, inform efferent units that this unit has emitted an event, and update the related event-times *only* if this event-time is lower than its previous estimation,

-c- store the event-time lower-bound in order to re-ask the unit at that time.

A simple way to interpret this modification is to consider that a unit can generate “silent events” which write: “Ask me again at that time, I will better know”.

As soon as each unit is able to *provide the next event-time after a finite number of lower-bound estimations*, the previous process is valid.

This new paradigm is fully compatible with the original, in the sense that units simulated by the original mechanism are obviously simulated here, they simply never return lower-bounds.

It appears that the implementation of this “variant” in the simulation kernel is no more than a few additional line of codes. However, the specifications of an event-unit deeply change, since the underlying calculations can be totally different.

We refer to this modified paradigm as the *lazy* event-based simulation mechanism.

The reason of this change of paradigm is twofold:

- Event-based and clock-based calculations can be easily mixed using the same mechanism.

  Units that must be simulated with a clock-based mechanism simply return the next clock-tick as lower-bound, unless they are ready to fire an event. However in this case, each unit can choose its own clock, requires low-rate update when its state is stable or require higher-rate update when in transient stage, etc.. Units with different strategies can be mixed, etc..

  For instance, in [29] units corresponding to synapses are calculated in event-based mode, while units corresponding to the neuron body are calculated in clock-based mode, minimizing the overall computation load. They however use a more complicated specific mechanism and introduce approximations on the next spike-time calculations.

  At the applicative level, this changes the point of view with respect to the choice of event-based/clock-based simulations. Now, an event-based mechanism can always simulate clock-based mechanisms, using this useful trick.

- Computation time can be saved by postponing some calculations.

  Event-based calculation is considered as costless than clock-based calculation because the neuron state is not recalculated at each time-step, only when a new event is input. However, as pointed out by several authors, when a large amount of events arrive to a unit, the next event-time is recalculated a large amount of time which can be much higher than a reasonable clock rate, inducing a fall of performances.

  Here this drawback can be limited. When a unit receives an event, it does not need to recalculate the next event-time, as soon as it is known as lower that the last provided event-time bound. This means that if the input event is “inhibitory” (i.e. tends to increase the next event-time) or if the unit is not “hyper-polarized”

(i.e. not close to the firing threshold, which is not trivial to determine) the calculation can be avoided, while the opportunity to update the unit state again later is to be required instead.

Remarks

Mixing event-based and clock-based calculations that way is reasonable, only because the event-time queue retrieve/update operations have a very low cost. Otherwise, clock-ticks would have generated prohibitory time overloads.

Changing the event-based paradigm is not a simple trick and may require to reconsider the simulation of neural units. This is addressed in the sequel for two important classes of biologically plausible neural units, at the edge of the state of the art and of common use: Synaptic conductance based models [14] and spike response models [16].

3 Event-based simulation of adaptive non-linear gIF models

Let us consider a *normalized* and *reduced* "punctual conductance based generalized integrate and fire" (gIF) neural unit model [14] as reviewed in [38].

The model is normalized in the sense that variables have been scaled and redundant constants reduced. This is a standard usual one-to-one transformation, discussed in the next subsection.

The model is reduced in the sense that both *adaptive* currents and non-linear *ionic* currents are no more explicitly depending on the potential membrane, but on time and previous spikes only. This is a non-standard approximation and a choice of representation carefully discussed in appendix 3.1.

Let v be the normalized membrane potential and $\tilde{\omega}_t = \{\cdots t_i^n \cdots\}$ the list of all spike times $t_i^n < t$. Here t_i^n is the n-th spike-time of the neuron of index i. The dynamic of the integrate regime writes:

$$\frac{dv}{dt} + g(t, \tilde{\omega}_t)\, v = i(t, \tilde{\omega}_t), \tag{1}$$

while the fire regime (spike emission) writes $v(t) = 1 \Rightarrow v(t^+) = 0$ with a firing threshold at 1 and a reset potential at 0, for a normalized potential.

Equation (1) expands:

$$\frac{dv}{dt} + \frac{1}{\tau_L}\left[v - E_L\right] + \sum_j \sum_n r_j\left(t - t_j^n\right)\left[v - E_j\right] = i_m(\tilde{\omega}_t), \tag{2}$$

where τ_L and E_l are the membrane leak time-constant and reverse potential, while $r_j()$ and E_j the spike responses and reverse potentials for excitatory/inhibitory synapses and gap-junctions as made explicit in appendix A. Here, $i_m()$ is the reduced membrane current, including simplified adaptive and non-linear ionic current.

3.1 Reduction of internal currents

Let us now discuss choices of modeling for $i_m = I^{adapt} + I^{ionic}$

Adaptive current

In the Fitzhugh-Nagumo reduction of the original Hodgkin-Huxley model [18] the average kinematics of the membrane channels is simulated by a unique adaptive current. Its dynamics is thus defined, between two spikes, by a second equation of the form:

$$\tau_w \frac{dI^{adp}}{dt} = g_w\ (V - E_L) - I^{adp} + \Delta_w\, \delta\left(V - V_{\text{threshold}}\right),$$

with a slow time-constant adaptation $\tau_w \simeq 144ms$, a sub-threshold equivalent conductance $g_w \simeq 4nS$ and a level $\Delta_w \simeq 0.008nA$ of spike-triggered adaptation current. It has been shown [19] that when a model with a quadratic non-linear response is increased by this adaptation current, it can be tuned to reproduce qualitatively all major classes of neuronal in-vitro electro-physiologically defined regimes.

Let us write:

$$\begin{aligned} I^{adp}(V,t) &= e^{\frac{-(t-t_0)}{\tau_w}}\, I^{adp}(t_0) + \tfrac{g_w}{\tau_w} \int_{t_0}^{t} e^{\frac{-(t-s)}{\tau_w}} (V(s) - E_L) ds + \Delta_w\, \#(t_0,t) \\ &\simeq \underbrace{e^{\frac{-(t-t_0)}{\tau_w}}\, I^{adp}(t_0) + g_w \left(1 - e^{\frac{-(t-t_0)}{\tau_w}}\right) (\bar{V} - E_L)}_{slow\ variation} + \underbrace{\Delta_w\, \#(t_0,t)}_{spike-time\ dependent} \end{aligned}$$

where $\#(t_0,t)$ is the number of spikes in the $[t_0,t]$ interval while $\bar{V}$ is the average value between t and t_0, the previous spike-time.

Since the time-constant adaptation is slow, and since the past dependency in the exact membrane potential value is removed when resetting, the slow-variation term is almost constant. This adaptive term is mainly governed by the spike-triggered adaptation current, the other part of the adaptive current being a standard leak. This is also verified, by considering the linear part of the differential system of two equations in V and I^{adp}, for an average value of the conductance $\bar{G}^+ \simeq 0.3 \cdots 1.5nS$ and $\bar{G}^- \simeq 0.6 \cdots 2.5nS$. It appears that the solutions are defined by two decreasing exponential profiles with $\tau_1 \simeq 16ms << \tau_2 \simeq 115ms$ time-constants, the former being very close to the membrane leak time-constant and the latter inducing very slow variations.

In other words *current adaptation is, in this context, mainly due to spike occurrences* and the adaptive current is no more directly a function of the membrane potential but function of the spikes only.

Non-linear ionic currents

Let us now consider the non-linear active (mainly Sodium and Potassium) currents responsible for the spike generation. In models designed to simplify the complex structure of Hodgkin-Huxley equations, the sub-threshold membrane potential is defined by a supra-linear kinematics, taken as e.g. quadratic or exponential, the latter form closer to observed biological data [3]. It writes, for example:

$$I^{ion}(V) = \frac{C\,\delta_a}{\tau_L} e^{\frac{V-E_a}{\delta_a}} \geq 0 \text{ with } \left.\frac{dI^{ion}}{dV}\right|_{V=E_a} = \frac{C}{\tau_L} \tag{3}$$

with $E_a \simeq -40mV$ the threshold membrane state at which the slope of the I-V curve vanishes, while $\delta_a = 2mV$ is the slope factor which determines the sharpness of the threshold. There is no need to define a precise threshold in this case, since the neuron fires when the potential diverges to infinity.

A recent contribution [43] re-analyzes such non-linear currents, proposes an original form of the ionic current, with an important sub-threshold characteristic not present in previous models [19, 3] and show that one obtains the correct dynamics, provided that the profile is mainly non-negative and strictly convex. This is not necessarily a quadratic or exponential function.

Making profit of this general remark, we propose to use a profile of the form of (3), but simply freeze the value of V to the the previous value obtained at the last spike time occurrence. This allows us to consider a supra-linear profile which depends only on the previous spike times[2]. This approximation may slightly underestimate the ionic current before a spike, since V is increasing with time. However, when many spikes are input, as it is the case for cortical neurons, errors are minimized since the ionic current update is made at high rate.

At a phenomenological level [19] the real goal of this non-linear current in synergy with adaptive currents is to provide several firing regimes. We are going to verify experimentally that even coarser approximations allow to attain this goal.

3.2 Derivation of a spike-time lower-bound.

Knowing the membrane potential at time t_0 and the list of spike times arrival, one can obtain the membrane potential at time t, from (1):

$$v(t) = \nu(t_0, t, \tilde{\omega}_{t_0})\, v(t_0) + \int_{t_0}^{t} \nu(s, t, \tilde{\omega}_{t_0})\, i(s, \tilde{\omega}_{t_0})\, ds \tag{4}$$

[2] In fact a more rigorous result can be derived, although at the implementation level, the simple heuristic proposed here seems sufficient. Let us write $i(V, t, \tilde{\omega}_t) = i'(t, \tilde{\omega}_t) + I^{(ion)}(V, t, \tilde{\omega}_t)$ thus separate the $I^{(ion)}$ from all other currents written $i'(t, \tilde{\omega}_t)$. Let us consider the last spike time t_0 of this neuron and let us write $\tilde{V}$ the solution of the linear differential equation "without" the ionic current $I^{(ion)}$:

$$C\,\frac{d\tilde{V}}{dt} + g(t, \tilde{\omega}_t)\,\tilde{V} = i'(t, \tilde{\omega}_t)$$

with $V(t_0) = V_{reset}$, as obtained above. Define now $\hat{V} = V - \tilde{V}$, with $\hat{V}(t_0) = 0$, V being the solution of the previous equation (without the ionic current). This yields:

$$C\,\frac{d\hat{V}}{dt} + g(t, \tilde{\omega}_t)\,\hat{V} = I^{(ion)}(\hat{V} + \tilde{V}(t, \tilde{\omega}_t), t, \tilde{\omega}_t)$$

as easily obtained by superposition of the linear parts of the equation.
Let $h(t, \tilde{\omega}_t)$ be any regular function and $f(V)$ any bijective regular function with $f(\hat{V}) \neq 0$. These two functions allow us to model a whole family of ionic currents:

$$I^{(ion)}(\hat{V} + \tilde{V}(t, \tilde{\omega}_t), t, \tilde{\omega}_t) = g(t, \tilde{\omega}_t)\,\hat{V} + \frac{h(t, \tilde{\omega}_t)}{f(\hat{V})}.$$

The choice of h and f is simply related to specific properties: The reader can easily verify, by a simple integration, that it allows to obtain a closed form:

$$\hat{V}(t, \tilde{\omega}_t) = F^{-1}\left(\int_{t_0}^{t} h(s, \tilde{\omega}_t)\, ds\right) \text{ with } F' = f \text{ and } F(0) = 0.$$

so that $\hat{V}$ is now a function of $\tilde{\omega}_t, t$ with $\hat{V}(t_0, \tilde{\omega}_t) = 0$, and so is $I^{(ion)}(\hat{V}(t, \tilde{\omega}_t) + \tilde{V}(t, \tilde{\omega}_t), t, \tilde{\omega}_t)$, removing the direct dependence on V. In other words, it now depends only on t and on the spike times (thus on $\tilde{\omega}_t$) and not anymore on the membrane potential explicitly. Clearly, this only applies to neurons which have fired at least once during the period of observation. Otherwise, we assume that its initial condition was also V_{reset}.
We can, .e.g, choose:

$$\begin{aligned} I^{(ion)}(V, t, \tilde{\omega}_t) &= \frac{C\,\delta_a}{\tau_L} e^{\frac{V(t) - E_a(t, \tilde{\omega}_t)}{\delta_a}} \\ E_a(t, \tilde{\omega}_t) &= \tilde{V}(t, \tilde{\omega}_t) - \delta_a\, ln\left(\frac{g(t, \tilde{\omega}_t)}{\bar{g}}\right) \end{aligned}$$

for any $\bar{g} > 0$ which allows to control the threshold for different conductance.
Here $h = g$ and $f(v) = (k\, e^{\frac{v}{\delta_a}} - v)^{-1}$ for some k.
In this case the threshold is no more fixed, but adaptive with respect to $g(t)$: the higher the conductance, the higher the threshold (via the $\tilde{V}$). This is coherent with what has been observed experimentally [2, 49], since the higher the conductance, the higher the frame rate increases with the spiking threshold.

with:

$$\log(\nu(t_0,t_1,\tilde{\omega}_{t_0})) = -\int_{t_0}^{t_1} g(s,\tilde{\omega}_{t_0})\,ds \tag{5}$$

Furthermore,

$$\begin{array}{rcll} g(t,\tilde{\omega}_{t_0}) & = & \frac{1}{\tau_L} + \sum_j \sum_n r_j\left(t-t_j^n\right) & > \quad 0 \\ i(t,\tilde{\omega}_{t_0}) & = & \frac{1}{\tau_L} E_L + \sum_j \sum_n r_j\left(t-t_j^n\right)\, E_j + i_m(\tilde{\omega}_{t_0}) & \geq \quad 0 \end{array} \tag{6}$$

since the leak time-constant, the conductance spike responses are positive, the reverse potential are positive (i.e. they are larger than or equal to the reset potential) and the membrane current is chosen positive.

The spike-response profile schematized in Fig. 1 and a few elementary algebra yields to the following bounds:

$$t \in [t_0,t_1] \Rightarrow r_\wedge(t_0,t_1) \leq r(t) \leq r_\vee(t_0,t_1) + t\, r'_\vee(t_0,t_1) \tag{7}$$

writing r' the time derivative of r, with

$$r_\wedge(t_0,t_1) = \min(r(t_0),r(t_1)) \text{ and } r_\vee(t_0,t_1) = \max(r(t_0),r(t_1))$$

with a similar definition for $r'_\vee(t_0,t_1)$.

Here we thus consider a constant lower-bound $r_\wedge$ and a linear or constant upper-bound $r_\vee + t\, r'_\vee$. The related two parameters are obtained considering in sequence the following cases:

	$t_0 \in$	$t_1 \in$	$r_\vee(t_0,t_1)$	$r'_\vee(t_0,t_1)$
(i)	$[t_1 - r(t_1)/r'(t_1), t_a]$	$[t_a,t_b]$	$r(t_1) - t_1\, r'(t_1)$	$r'(t_1)$
(ii)	$[t_a,t_b]$	$[t_0,+\infty]$	$r(t_0) - t_0\, r'(t_0)$	$r'(t_0)$
(iii)	$[t_c,+\infty]$	$]t_0,+\infty]$	$(r(t_0)\, t_1 - r(t_1)\, t_0)/(t_1-t_0)$	$(r(t_1)-r(t_0))/(t_1-t_0)$
(iv)	$]-\infty,t_1]$	$[t_0,t_b]$	$r(t_1)$	0
(v)	$]-\infty,t_b]$	$[t_b,+\infty]$	$r(t_b)$	0
(vi)	$[t_b,t_1]$	$[t_0,+\infty]$	$r(t_0)$	0

In words, conditions (i) and (ii) correspond to the fact that the the convex profile is below its tangent (schematized by d in Fig. 1), condition (iii) that the profile is concave (schematized by d'' in Fig. 1). In other cases, it can be observed that 1st order (i.e. linear) bounds are not possible. We thus use constant bounds (schematized by d' in Fig. 1). Conditions (iv) and (vi) correspond to the fact the profile is monotonic, while condition (v) corresponds to the fact the profile is convex. Conditions (iv), (v) and (vi) correspond to all possible cases. Similarly, the constant lower bound corresponds to the fact the profile is either monotonic or convex.

From these bounds we derive:

$$\begin{array}{rcl} g(t,\tilde{\omega}_{t_0}) & \geq & \frac{1}{\tau_\wedge(t_0,t_1)} \\ & \stackrel{\text{def}}{=} & \frac{1}{\tau_L} + \sum_j \sum_n r_{j\wedge}(t_0,t_1) \\ i(t,\tilde{\omega}_{t_0}) & \leq & i_\vee(t_0,t_1) + t\, i'_\vee(t_0,t_1) \\ & \stackrel{\text{def}}{=} & \left[\frac{1}{\tau_L} E_L + \sum_j \sum_n r_{j\vee}(t_0,t_1)\, E_j + i_m(\tilde{\omega}_{t_0})\right] \\ & + & t\, \left[\sum_j \sum_n r'_{j\vee}(t_0,t_1)\, E_j\right] \end{array} \tag{8}$$

while $i'_\vee(t_0,t_1) \geq 0$ as the positive sum of $r'_{j\vee}$ values is always positive in our case.

Combining with (4), since values are positive, yields:

$$v(t) \leq v_\vee(t) \stackrel{\text{def}}{=} (v(t_0) - v_\circ)\, e^{-\frac{t-t_0}{\tau_\wedge}} + v_\bullet + i_\bullet\, t \tag{9}$$

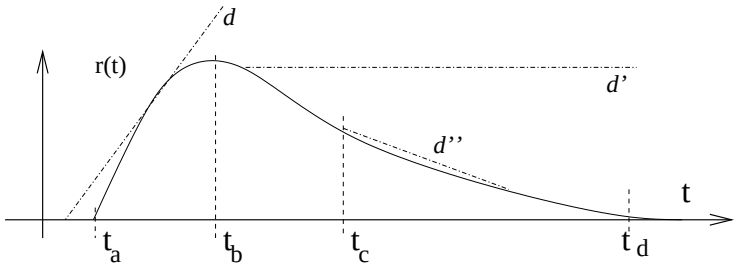

Figure 1: The spike response profile. It has a flat response during the absolute delay interval $[0, t_a]$, an increasing convex profile until reaching its maximum at t_b, followed by a decreasing convex and then concave profile, with an inflexion point at t_c. After t_d the response is negligible. See text for details about d, d' and d''.

writing:

$$\begin{array}{rcl} i_\bullet & \stackrel{\text{def}}{=} & \tau_\wedge(t_0,t_1)\, i'_\vee(t_0,t_1) \\ v_\bullet & \stackrel{\text{def}}{=} & \tau_\wedge(t_0,t_1)\,(i_\vee(t_0,t_1) - \tau_\wedge(t_0,t_1)\, i'_\vee(t_0,t_1)) \\ v_\circ & \stackrel{\text{def}}{=} & v_\bullet + i_\bullet\, t_0 \end{array} \tag{10}$$

Finally we can solve the equation for $t^\vee \stackrel{\text{def}}{=} v_\vee^{-1}(1)$ and obtain:

$$t^\vee(t_0,t_1) = \begin{cases} t_0 & , \quad v(t0) \geq 1 \\ \frac{1-v_\bullet}{i_\bullet} + \tau_\wedge \,\mathrm{L}\left(\frac{v_\circ - v(t_0)}{\tau_\wedge\, i_\bullet}\, e^{\frac{v_\circ - 1}{\tau_\wedge\, i_\bullet}}\right) & , \quad i'_\vee(t_0,t_1) > 0 \\ t_0 + \tau_\wedge \,\log\left(\frac{v_\bullet - v(t_0)}{v_\bullet - 1}\right) & , \quad i'_\vee(t_0,t_1) = 0, v_\bullet > 1 \\ +\infty & , \quad \text{otherwise} \end{cases} \tag{11}$$

Here $y = \mathrm{L}(x)$ is the Lambert function defined as the solution, analytic at 0, of $y\, e^y = x$ and is easily tabulated.

The derivations details are omitted since they have been easily obtained using a symbolic calculator.

In the case where $i'_\vee(t_0,t_1) = 0$ thus $i_\bullet = 0$, $v_\vee(t)$ corresponds to a simple leaky integrate and fire neuron (LIF) dynamics and the method thus consists of upper bounding the gIF dynamics by a LIF in order to estimate a spiking time lower-bound. This occurs when constant upper-bounds is used for the currents. Otherwise (9) and (10) corresponds to more general dynamics.

3.3 Event-based iterative solution

Let us apply the previous derivation to the calculation of the next spike-time lower-bound for a gIF model, up to a precision ϵ_t. One sample run is shown Fig. 2

Given a set of spike-times t_j^n and an interval $[t_0, t_1]$, from (8) $\tau_\wedge$, $i_\vee$ and $i'_\vee$ are calculated in about $10^1\, M\, C$ operations, for an average of C connections per units, and with an average number $M \leq N$ of firing units. This is the costly part of the calculation[3], and is equivalent to a single clock-based integration step. Spike response profiles

[3]It appears, that since each synapse response corresponds to a linear differential equation which could be analytically integrated, the global synaptic response $\bar{r}_j(t) = \sum_n r_j(t - t_j^n)$ can be put in closed form, and then bounded by constant values, thus reducing the computation complexity from $O(MC)$ to $O(C)$

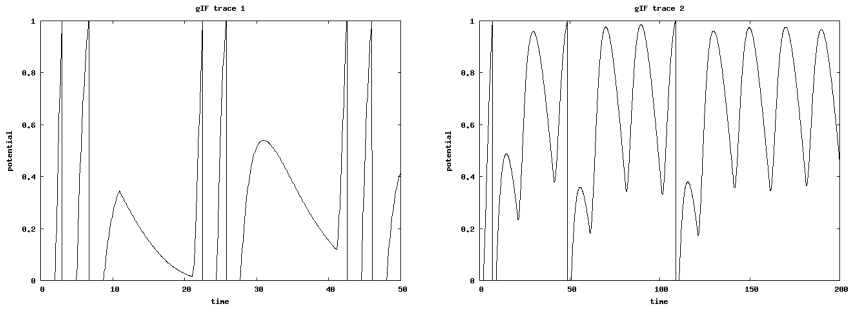

Figure 2: An example of gIF normalized membrane potential. The left trace corresponds to $50ms$ of simulation, the neuron being artificially connected to a periodic excitatory/inhibitory input neuron pair at 50/33 Hz of high synaptic weight, in order to make explicit the double exponential profiles. The right trace corresponds to $200ms$ of simulation, with a higher inhibition. The weights have been chosen, in order to make the neuron adaptation explicit: the firing frequency decreases until obtaining a sub-threshold membrane potential.

$r_j()$ and profile derivatives $r'_j()$ for excitatory/inhibitory synapses and gap junctions are tabulated with a ϵ_t step. Then, from (10) and (11) we obtain $t^\vee(t_0, t_1)$.

The potential $v(t_0)$ is calculated using any well-established method, as detailed in, e.g., [37], and not reviewed here.

The following algorithm guarantees the estimation of a next spike-time lower-bound after t_0. Let us consider an initial interval of estimation d, say $d \simeq 10ms$:

-a- The lower-bound $t_\vee = t^\vee(t_0, t_0 + d)$ is calculated.
-b- If $t_0 + d \leq t_\vee$ the lower-bound value $t_0 + d$ is returned
the estimation interval is doubled $d \leftarrow 2\,d$
-c- If $t_0 + \epsilon_t < t_\vee < t_0 + d$ the lower-bound value $t_\vee$ is returned
the estimation interval is reduced $d \leftarrow 1/\sqrt{2}\,d$
-d- If $t_0 < t_\vee < t_0 + d$, $d \leq \epsilon_t$, the next spike-time $t_\vee$ is returned.

Step -b- corresponds to the case where the neuron is not firing in the estimation interval. Since $v(t)$ is bounded by $v_\vee(t)$ and the latter is reaching the threshold only outside $[t_0, t_0 + d]$, $t_0 + d$ is a time lower-bound. In addition, a heuristic is introduced, in order to increase the estimation interval in order to save computation steps.

Step -c- corresponds to a strict lower-bound computation, with a relative value higher than the precision ϵ_t.

Step -d- assumes that the lower-bound estimation converges towards the true spike-time estimation when $t_1 \to t_0$.

This additional convergence property is easy to derive. Since:
$\lim_{t_1 \to t_0} r_{j\wedge}(t_0, t_1) = \lim_{t_1 \to t_0} r_{j\vee}(t_0, t_1) = r_j(t_0), \lim_{t_1 \to t_0} r'_{j\vee}(t_0, t_1) = r'_j(t_0)$,
then:
$\lim_{t_1 \to t_0} 1/\tau_\wedge(t_0, t_1) = g(t_0), \lim_{t_1 \to t_0} i_\vee(t_0, t_1) = i(t_0), \lim_{t_1 \to t_0} i'_\vee(t_0, t_1) = i'(t_0)$,
yielding:

$$\lim_{t_1 \to t_0} v_\vee(t) = \bar{v}_\vee(t) \stackrel{\text{def}}{=} (v(t_0) - \bar{v}_\circ)\, e^{-g(t_0)\,(t - t_0)} + \bar{v}_\bullet + \bar{i}_\bullet\, t$$

as detailed in [37]. This well-known issue is not re-addressed here, simply to avoid making derivations too heavy.

writing:

$$\begin{aligned} \bar{i}_\bullet &\stackrel{\text{def}}{=} 1/g(t_0)\, i'(t_0) \\ \bar{v}_\bullet &\stackrel{\text{def}}{=} 1/g(t_0)\,(i(t_0) - 1/g(t_0)\, i'(t_0)) \\ \bar{v}_\circ &\stackrel{\text{def}}{=} \bar{v}_\bullet + \bar{i}_\bullet\, t_0 \end{aligned}$$

We thus obtain a limit expression $\bar{v}_\vee$ of $v_\vee$ when $t_1 \to t_0$. From this limit expression we easily derive:

$$v(t) - \bar{v}_\vee(t) = -1/2\, g'(t_0)\, v(t_0)\, t^2 + O(t^3),$$

and finally obtain a quadratic convergence, with an error closed-form estimation.

The methods thus corresponds to a semi-interval estimation methods of the next spike-time, the precision ϵ_t being adjustable at will.

The interval of estimation d is adjusted by a very simple heuristic, which is of standard use in non-linear numerical calculation adjustment.

The unit calculation corresponds to one step of the iterative estimation, the estimation loop being embedded in the simulator interactions. This is an important property as far as real-time computation is concerned, since a minimal amount of calculation is produced to provide, as soon as possible, as suboptimal answer.

This "lazy" evaluation method is to be completed by other heuristics:
- if the input spike is inhibitory thus only delaying the next spike-time, re-calculation can be avoided;
- if all excitatory contributions $g_\vee = M\, r^+(t_b)$ are below the spiking threshold, spike-time is obviously infinity;
- after a spike the refractory period allows us to postpone all calculation (although synaptic conductances still integrate incoming spikes).

In any case, comparing to other gIF models event-based simulation methods [5, 6, 38], this alternative method allows one to control the spike-time precision, does not constraint the synaptic response profile and seems of rather low computational cost, due to the "lazy" evaluation mechanisms.

Numerical convergence of the lower-bound iteration

Considering biologically plausible parameters as reviewed here and in appendix A, we have experimented carefully the numerical convergence of this lower-bound iterative estimation, considering a gIF neuron with adaptive and non-linear internal currents, implemented as proposed here, and providing membrane potential, e.g., as shown in Fig. 2.

Let us report our numerical experimentation.

We have always observed the convergence of the method (also extensively experimented at the network level in a next section), with a convergence in about $2 - 20$ iterations (mean $\simeq 11$, standard-deviation $\simeq 5$), the lower-bound iteration generating steps of about $0.01 - 10ms$ (mean $\simeq 3ms$, standard-deviation $\simeq 4ms$) from one lower-bound to another, with three distinct qualitative behaviors:
-a- Sub-threshold maximal potential: the previous calculation estimates a maximal membrane potential below the threshold and calculation stops, the neuron being quiet; in this mode the event-based strategy is optimal and a large number of calculations are avoided with respect to clock-based paradigms.
-b- Sub-threshold lower-bound estimation: the maximal membrane potential is still estimated over the threshold, with a next-spike time lower bound. In this mode, we observed an exponential increase of the next-spike time lower bound and in $2 - 5$ iterations the maximal membrane potential estimation is estimated under the threshold,

switching to mode -a-; in this mode the estimation interval heuristic introduced previously is essential and the next-spike time lower bound estimation allows the calculation to quickly detect if the neuron is quiet.
-c- Iterative next-spike time estimation: if a spike is pending, the previous calculation estimates in about $10 - 20$ iterations the next-spike time up to a tunable precision (corresponding to the dt of the simulation mechanism). The present mechanism acts as an iterative estimation of the next spike time, as expected.

4 About event-based simulation of SRM models

Among spiking-neuron models, the Gerstner and Kistler spike response model (SRM) [16] of a biological neuron defines the state of a neuron via a single variable:

$$u_i(t) = r_i\, j(t) + \nu_i(t - t_i^*) + \sum_j \sum_{t_j^n \in \mathcal{F}_j} w_{ij}\, \varepsilon_{ij}(t - t_i^*, (t - t_j^n) - \delta_{ij}) \qquad (12)$$

where u_i is the normalized membrane potential, $j()$ is the continuous input current for an input resistance r_i. The neuron fires when $u_i(t) \geq \theta_i$, for a given threshold, ν_i describes the neuronal response to its own spike (neuronal refractoriness), t_i^* is the last spiking time of the ith neuron, ε_{ij} is the synaptic response to pre-synaptic spikes at time t_j^n post-synaptic potential (see Fig. 3), w_{ij} is the *connection strength* (excitatory if $w_{ij} > 0$ or inhibitory if $w_{ij} < 0$) and δ_{ij} is the *connection delay* (including axonal delay). Here we consider only the last spiking time t_i^* for the sake of simplicity, whereas the present implementation is easily generalizable to the case where several are taken into account.

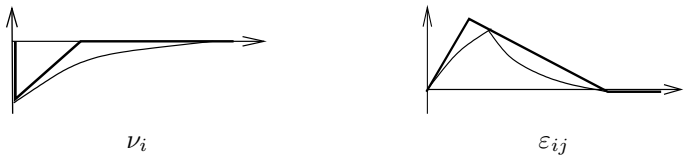

Figure 3: Potential profiles used in equation (12). The original exponential profiles derived by the authors of the model are shown as thin curves and the piece-wise linear approximation as thick lines. Any other piece-wise linear profiles could be considered, including piece-wise finer linear approximation of exponential profiles.

Let us call here lSRM such piece-wise linear approximations of SRM models.

This model is very useful both at the theoretical and simulation levels. At a computational level, it has been used (see [26] for a review) to show that any feed-forward or recurrent (multi-layer) analog neuronal network (à-la Hopfield, e.g., McCulloch-Pitts) can be simulated arbitrarily closely by an insignificantly larger network of spiking neurons, even in the presence of noise, while the reverse is not true [24, 25]. In this case, inputs and outputs are encoded by temporal delays of spikes. These results highly motivate the use of spiking neural networks.

This lSRM model has also been used elsewhere (see e.g. [41] for a review), including high-level specifications of neural network processing related to variational approaches [45], using spiking networks [46]. The authors used again a lSRM to implement their non-linear computations.

Let us make explicit here the fact a lSRM can be simulated on an event-based simulator for two simple reasons:
- the membrane potential is a piece-wise linear function as the sum of piece-wise linear functions (as soon as the optional input current is also piece-wise constant or linear),
- the next spike-time calculation is obvious to calculate on a piece-wise linear potential profile, scanning the linear segments and detecting the 1st intersection with $u = 1$ if any. The related piece-wise linear curve data-structure has been implemented[1] and support three main operations:
- At each spike occurrence, add linear pieces to the curve corresponding to refractoriness or synaptic response.
- Reset the curve, after a spike occurrence.
- Solve the next spike-time calculation.
This is illustrated in Fig. 4.

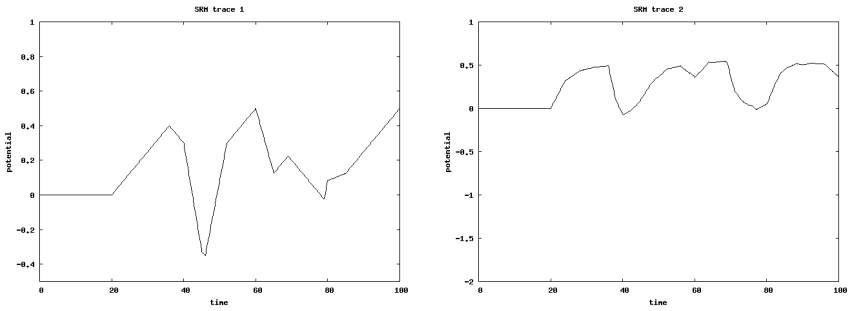

Figure 4: An example of lSRM normalized membrane potential. The traces corresponds to $100ms$ of simulation. The leftward trace uses the fastest possible piece-wise linear approximation of the SRM model profiles. The rightward trace is simulated with a lower excitation inhibition and using a thinner piece-wise linear approximation. The neuron is defined by biologically plausible parameters as reviewed in appendix A . It is connected to a pair of periodic excitatory and inhibitory input neurons, with different time constants, as in Fig. 2.

This is to be compared with other simulations (e.g. [28, 40]) where stronger simplifications of the SRM models have been introduced to obtain a similar efficiency, whereas other authors propose heavy numerical resolutions at each step. When switching from piece-wise linear profiles to the original exponential profiles [16], the equation to solve is now of the formal form:

$$1 = \sum_{i=1}^{n} \lambda_i e^{t/\tau_i},$$

without any closed-form solution as soon as $n > 1$. One elegant solution [6] is to approximate the time-constant by rational numbers and reduce this problem to a polynomial root finding problem. Another solution is to upper-bound the exponential profiles by piece-wise linear profiles in order to obtain a lower-bound estimation of the next spike-time and refine in the same way as what has been proposed in the previous section. Since the mechanism is identical, we are not going to further develop.

In any case, this very powerful phenomenological model of biological dynamics can be simulated with several event-based methods, including at a fine degree of precision, using more complex piece-linear profiles.

5 Experimental results

5.1 Kernel performances and features

In order to estimate the kernel sampling capability we have used, as a first test, a random spiking network with parameter less connections, the spiking being purely random thus not dependent on any input.

In term of performances, on a standard portable computer (Pentium M 750 1.86 GHz, 512Mo of memory) we process about 10^{5-7} spike-time updates / second, given the network size and connectivity. Performances reported in Fig. 5 confirm that the algorithmic complexity only marginally depends on the network size, while it is mainly function of the number of synapses (although both quantities are indeed linked). We also notice the expected tiny overhead when iterating on empty boxes in the histogram, mainly visible when the number of spikes is small. This overhead is constant for a given simulation time. The lack of proportionality in performances is due to the introduction of some optimization in the evaluation of spike-times, which are not updated if unchanged.

We have also observed that the spike-time structure upper and lower bounds D and dt have only a marginal influence on the performances, as expected.

Moreover, the numbers in Fig. 5 allows to derive an important number: the overhead for an event-based implementation of a clock-based mechanism. Since we can process about $2\,10^6$ updates/second while we have measured independently that a minimal clock-based mechanism process about $5\,10^7$ updates/second, we see that the cost the overhead is of about $0.5\mu s$ / update. This number is in coherence with the number of operations to realize at time modification in the underlying data structure.

It is important to clarify these apparently "huge" performances. The reason is that the event-based simulation kernel is *minimal*. As detailed in Table 1, the implementation make a simple but extensive use of the best mechanisms of object oriented implementations. The network mechanism (i.e., the kernel) corresponds to about 10Kb of C++ source code, using a $\mathcal{O}(D/dt + N)$ buffer size and about $\mathcal{O}(1 + C + \epsilon/dt) \simeq 10 - 50$ operations/spike, for a size N network with C connections in average, while $\epsilon \ll 1$. This ϵ corresponds to the overhead when iterating on empty boxes in the histogram. We can use such a simple spike-time data structure because of the temporal constraints taken into account in our specifications.

As a consequence, not all spiking mechanisms are going to be simulated with this kernel: units with event-time intervals or input/output event delay below dt are going to generate a fatal error; units with inter-event intervals higher than D are also going to defeat this mechanism (unless an extension of the present mechanism, discussed previously, is not implemented). Note that despite these limitations the event-time accuracy itself is the not dt but the floating point machine precision.

Clock-based sampling in event-based environment. A step further, we have implemented a discretized version of a gIF network, called BMS, as detailed in [9, 11] (equations not reported here). The interest of this test is the fact that we can compare spike by spike an event-based and clock-based simulation since the latter is well-defined, thus without any approximation with respect to the former (see [11] for details).

We have run simulation with fully connected networks of size, e.g. $N = 100 - 1000$ over observation periods of $T = 1000 - 10000$ clocks, with the same random initial conditions and the same randomly chosen weights, as show in Fig.6. We have observed:

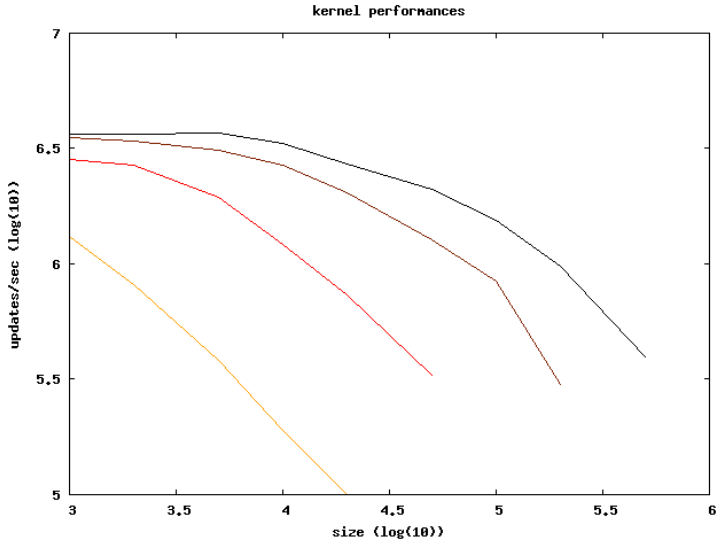

Figure 5: Simulation performances, for 10^5 spike firings and about $0.1s$ of simulation time. The CPU time is about $2s$ for 2^{13} neurons and 2^{21} synapses and does not depend on D or dt values, as expected. Spike-time insertion/deletion are counted as elementary updates. The network size in abscissa varies from 10^3 to about 10^6 and the number of connections from from 0 to 10^8, corresponding to curve end-points. Curves are shown for a connection probability of $P = 0$ (black, upper curve), $P = 10^{-3}$ (brown), $P = 10^{-2}$ (red), $P = 10^{-1}$ (orange). The performance is mainly function of the number of synapses, and marginally of the number of neurons.

```
template <class C> class Unit {
// Gets the next alter time:  event time or a lower-bound of the next
event time.
inline virtual double getNext(double present-time);

// Called to update the unit state, when the next spiking time or a
lower-bound occurs.
// Returns true if an event occurs, false it was a lower-bound
inline virtual bool next(double present-time);

// Called when an input unit fires an event.
// Returns true if the next alert time is to be updated, false otherwise.
inline virtual bool add(int neuron-index, C& connection-parameter, double
present-time);
};
```

Table 1: Specification of an event-based unit (pseudo-code). Each unit (neuron or group of neurons) specifies is next "alert" time and informs the network about event-occurrence. Lazy evaluation is implemented, at this level, via the fact that alert time is optional updated when receiving an event. The connection is templated in order for the kernel to be optimally recompiled for each kind of connection, while unit's mechanisms are inlined, allowing the compiler to eliminate code interface. The connection parameters is passed by reference, in order adaptation mechanisms to be implemented. See text for further details.

-1- the same raster (i.e., with a Victor-Purpura distance of 0 [44]); this exactitude is not surprising despite the fact that floating point errors accumulate[4]: we are performing

[4] Note that even if time is discretized, for BMS networks, the dynamics is based on floating point calculations, thus floating point errors accumulate. However as soon as spike is fired, the potential is reset and

the same floating point calculations in both cases, since the event-based implementation is exact, thus . . with the same errors;

-2- the overhead of the event-based implementation of the clock-based sampling is negligible (we obtain a number $< 0.1\mu s/step$), as expected. Again this surprisingly slow number is simply due to the minimal implementation, based on global time constraints, and the extensive use of the C/C++ optimization mechanisms.

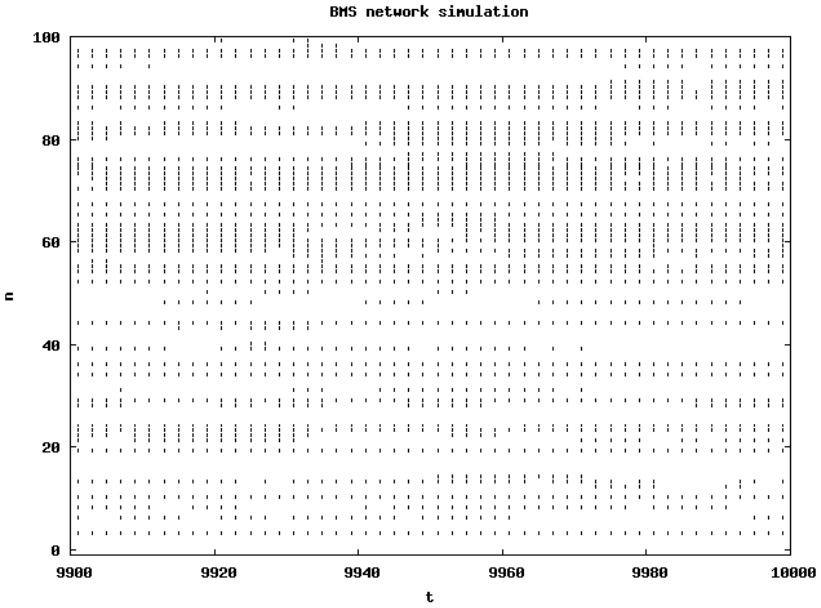

Figure 6: An example of BMS neural network simulation used to evaluate the clock-based sampling in the event-based kernel, for $N = 1000$ and $T = 10000$ thus 10^8 events. The first 100 neurons activity is shown at the end of the simulation. The figure simply shows the strong network activity with the chosen parameters, and .

Kernel usage. A large set of research groups in the field have identified what are the required features for such simulation tools [4]. Although the present implementation is *not* a simulator, but a simple simulator plug-in, we have made the exercise to list to which extends what is proposed here fits with existing requirements, as detailed in Table 3. We emphasize the fact the required programming is very light, for instance a "clock" neuron (allowing to mix clock-based and event-based mechanisms) writes:

```
class ClockUnit :  public Unit<bool> {
ClockUnit(double DT) : DT(DT), t(DT)   double DT, t;
inline virtual double getNext(double present-time)  return t; ;
inline virtual bool next(double present-time)  t += DT; return true; ;
inline virtual bool add(int neuron-index, bool connection-parameter,
double present-time)  return false; ;
};
```

providing DT is the sampling period. It is not easy to make things simple, and several possible choices of implementations have been investigated before proposing the interface proposed in Table 1.

previous errors are canceled. This explains why time-discretized simulations of IF networks are numerically rather stable.

See [35] for a further description of how event-based spiking neuron mechanisms can be implemented within such framework. Although presented here at a very pragmatical level, note that these mechanisms are based on the modular or hierarchical modeling strategy borrowed from the DEVS formalism (see, e.g., [35]).

5.2 Experimenting reduced adaptive and ionic currents

In order to experiment about our proposal to reduce ionic and adaptive currents to a function depending only on spike time, we consider a very simple model whose evolution equation at time `t` for the membrane potential `v` is:

```
if (t = 0)
                    v = 0; u = 0; t_0 = 0;
else if (v ≥ 1)
                    v = 0; u = u + k; t_0 = t;
else
                    v̇ = -g (v - E) - u + i;
                    if (t > t_0 + d) v = 0
```
(13)

where `u` is the adaptive current (entirely defined by equation (13)), `t_0` the last spiking time, d the non-linear current delay. The differential equation is simulated using an Euler interpolation as in [19, 43] to compare our result to what has been obtained by the other authors. The obvious event-based simulation of this model has been also implemented[1]. The input current `i` is either a step or a ramp as detailed in Fig. 7 and Table 2.

Four parameters, the constant leak conductance `g`, the reversal potential `E`, the adaptation current `k` step and the (eventually infinite) non-linear current delay `d` allows to fix the firing regime. These parameters are to be recalculated after the occurrence of each internal or external spike. In the present context, it was sufficient to use constant value except for one regime, as made explicit in Fig. 2. We use the two-stages current whose action is to reset the membrane current after a certain delay. We made this choice because it was the simplest and leads to a very fast implementation.

Experimental results are given in Fig. 7 for the parameters listed in Fig. 2. These results correspond to almost all well-defined regimes proposed in [19]. The parameter adjustment is very easy, we in fact use parameters given in [19] with some tiny adjustments. It is an interesting numerical result: The different regimes are generated by parameters values closed to those chosen for the quadratic model, the dynamic phase diagram being likely similar. See [43] for a theoretic discussion.

This places a new point on the performance/efficiency plane proposed by [20] at a very challenging place, and see that we can easily simulate different neuronal regimes with event-based simulations.

However, it is clear that such model does *not* simulates properly the neuron membrane potential as it is the case for the exponential model [3]. It is usable if and only if spike emission is considered, whereas the membrane potential value is ignored.

5.3 Experimental benchmarks

We have reproduced the benchmark 4 proposed in Appendix 2 of [4] which is dedicated to event-based simulation: it consists of 4000 IF neurons, which $80/20\%$ of excitatory/inhibitory neurons, connected randomly using a connection probability of $1/32 \simeq 3\%$. So called “voltage-jump” synaptic interactions are used: the membrane

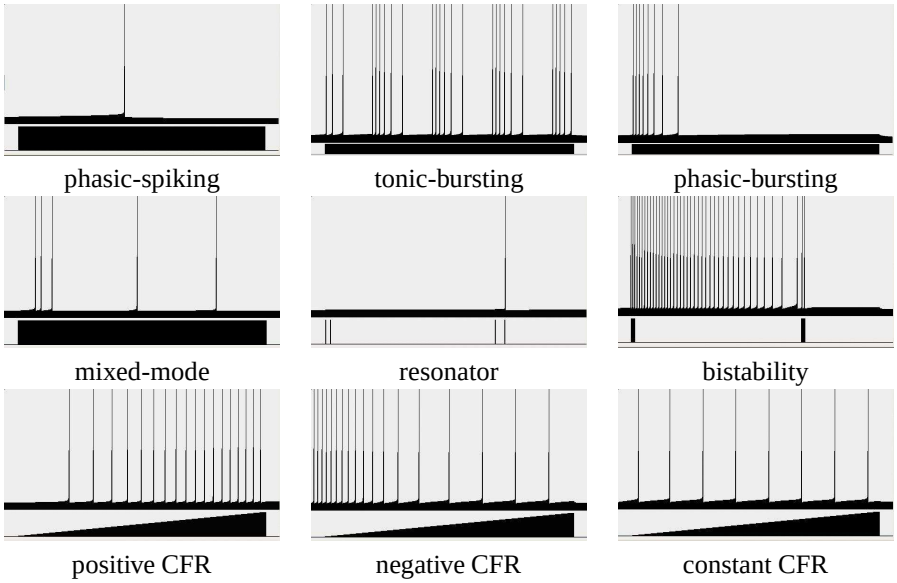

Figure 7: Typical results showing the versatility of the reduced model for spiking, bursting and other modes, including and different current-frequency-responses (CFR). For each mode, the upper trace shows the action potentials, the lower trace the input current. These results include the excitatory mode of type I where the spike frequency can be as small as possible in a $1 - 10^3 Hz$ range and of type II where the spike frequency remains bounded. Tonic spiking is not shown, since obvious to obtain.

spiking mode	leak conductance `g`	reverse potential `E`	adaptation step `k`	non-linear delay `d`	input magnitude `i(t)`	input form
phasic-spiking	0.04	0	30	$+\infty$	0.5	step
tonic-bursting	0.18	1.6	14.6	60	15	step
phasic-bursting	0.06	11	11.2	$+\infty$	0.5	step
mixed-mode	0.01	0	K	150	10	step
resonator	0.04	-27	0	$+\infty$	38	bi-pulse
bistability	0.88	80	1.8	$+\infty$	65	pulse
positive CFR	0.01	0	0	$+\infty$	30	ramp
negative CFR	0.52	80	4	$+\infty$	30	ramp
constant CFR	0.52	0	4	100	30	ramp

Table 2: Examples of parameters used to generate the spiking modes shown in Fig. 7. The mixed mode is simulated by a variable adaptation step $k = \{-20, 20\}$.

potential is abruptly increased/decreased by a value of $0.25/2.25mV$ for each excitatory/inhibitory event (thus using fixed randomly chosen weights). Here, we also introduce a synaptic delay of $2/4ms$ respectively and an absolute refractory period of $1ms$, both delays being corrupted by an additive random noise of $10\mu s$ of magnitude. We also have increased the network size and decreased the connection probability to study the related performances. In this network a synapse is simply defined by an index, weights are constant. See [4] for further details. One result is proposed in Fig. 8 to qualitatively verify the related network dynamics. The fact we find small inter-spike intervals in this case is coherent with previous observed results.

A step further, we also made profit of the new proposed approximation of gIF neuron models to run another test, inspired by another benchmark proposed in [4], after

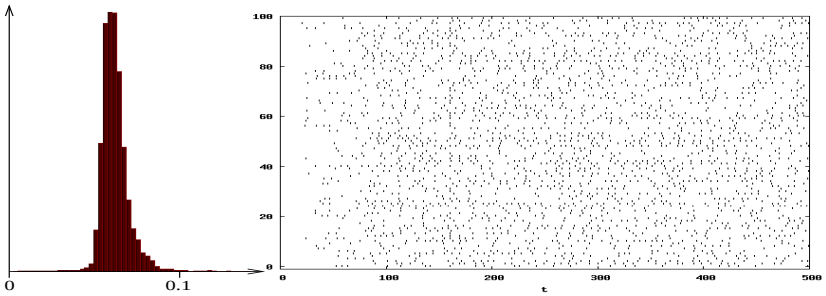

Figure 8: Inter-spike interval histogram (left view) in linear coordinates measured after $t > 0.1s$, and corresponding raster plot (right-view) during $1s$ of simulation, for the benchmark 4 proposed in Appendix 2 of (Brette etal, 2007).

[47], considering current-based interactions (CUBA model) and/or conductance-based interactions (COBA model). In our context, current based interactions correspond to gap junctions, while conductance-based interactions correspond to synaptic junctions. It was useless to reproduce the original benchmarks in [4] or [47], but interesting to experiment if we can explore the network dynamics with the improved model proposed here, using the method proposed in 3.1, and the parameters reviewed in appendix A, thus beyond CUBA/COBA models.

One result is shown in Figs. 9. Results are coherent with what is discussed in details in [47], and in particular close to what has been reviewed in [48]. This is clearly a preliminary test and the influence, on the network dynamics, of thus alternate model is out of the scope of the present work, and a perspective for a further study.

6 Discussion

Taking global temporal constraints into account, it has been possible to better understand, at the simulation levels to which extends spiking mechanisms are bounded and simplified. At this simulation level, the challenge is to generate spike-trains corresponding to what is observed in biology or to what is required for computational processing, without the necessity to *precisely reproduce the internal neuron state*. This is a very important simplification when the goal is to switch from the neural scale to the network one.

The proposed mechanism is a complement of existing simulation tools [4] in the following quantitative and qualitative senses:

Quantitative complementarity

As a software module, it has been designed to be as fast as possible.

The cost for this choice is that a programmatic interface is required, while in order to be available on any platform with the fastest performances, a `C/C++` implementation is required (interfaces to other programming languages being available). The fact that it is also a “small kernel” allows us to target embedded applications: since computing with spikes is now a mature methodology, a tool to run such algorithms on various platforms (e.g., in robotics) or embedded systems (e.g., intelligent reactive devices) was required.

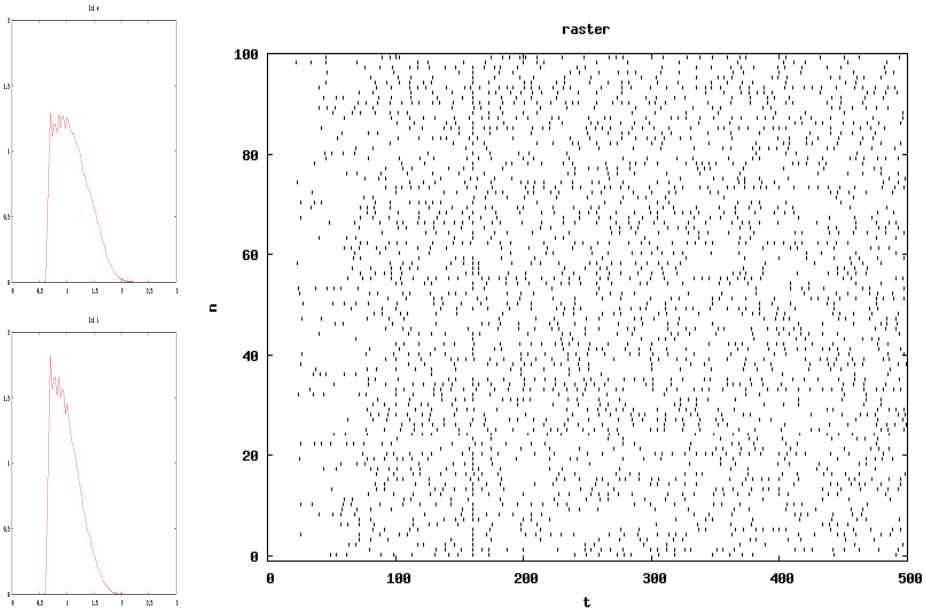

Figure 9: Inter-spike interval histogram for excitatory neurons (top-left) and inhibitory neurons (down-left) with the corresponding raster head (right). The abscissa is the decimal of the histogram log of the interval and the ordinate the inter-spike observed probability.

This has been possible here, without any loss in precision, the underlying data-structures being strongly simplified, but with another drawback: the network dynamics is constrained since spiking units must verify temporal constraints.

A step further, the use of models at the edge of the state of the art, such adaptive non-linear gIF networks, or SRM networks is made possible in an event-based framework, thus with expected better performances.

Regarding gIF networks, [6] has proposed a pure event-based method taking step-wise synapses with exponential decays into account, The same level of modeling has been proposed by [29], mixed with clock-based mechanisms, while [38] have investigated how to take synaptic alpha profiles into account. The proposed methods are based on sophisticated analytical derivations with the risk of having a rather huge number of operations to perform at each step. As a complementary variant of these methods, we propose here to introduce another degree of freedom, using iterative lower-bound estimations of the next spike time. This heuristic applied to gIF neurons seems to converge quickly in practice.

Regarding SRM networks, we have generalized the simple idea to use piece-wise linear profiles, approximating the original exponential profiles roughly (replacing the exponential curve by a line-segment) or at any level of precision (approximating the exponential curve by any number of line-segments). The precision/performance trade-off is thus adjustable at will.

The reason to consider gIF and SRM neuron simulation here is that they correspond, up to our best knowledge, to the most interesting punctual neuron models actually used in biologically plausible neural network simulation, in the deterministic case.

Qualitative complementarity

Two key points allow us to performs new simulations with this tool:

Event-based and clock-based mechanisms can be easily mixed here *in an event-based simulation mechanism*, whereas other implementations mix clock-based and event-based mechanisms in a clock-based simulator (e.g., in [29]), or use spike-time interpolation mechanisms in order to better approximate event-based mechanisms in such clock-based environment. Using an event-based simulator to simulate a clock is obvious, but usually stupid, because the event-based mechanism usually generates an heavy overhead, thus making the clock-based part of the simulation intractable. This is not the case here, since we use this minimal data-structure and have been able to see that the overhead is less than one micro-second on a standard laptop. It is thus appears a good design choice to use an event-based simulation mechanism to mixed both strategies.

The second key point is that, we have proposed a way to *consider adaptive non-linear gIF networks in an event-based framework*. It is easy to get convinced that 2D integrate and fire neurons differential equations with non-linear ionic currents (e.g. exponential, quartic or quadratic [42]) do not have closed-form solutions (unless in very special cases). Therefore, the next spike time is not calculable, except numerically, and the exact event-based implementation is not possible. Alternatives strategies have been proposed such as simulations with constant voltage steps [51] allowing to implement quadratic 1D (thus without adaptive currents) gIF networks in a modified event-base framework. In order to get rid of these limitations, one proposal developed here is to consider adaptive currents which depends only on the previous spiking time neuron state and non-linear ionic currents updated only at the each incoming spike occurrence. With these additional approximations, the event-based strategy can be used with such complex models. This is a complementary heuristic with respect to existing choices.

We notice that the present study only considers deterministic models, while the simulation of stochastic networks is also a key issue. Hopefully, event-based implementations of network of neurons with stochastic dynamics is a topic already investigated, both at the computer implementation level [35] and modeling level [34]. In the latter case, authors propose to reduce the multiple stochastic neuronal input activity to a dedicated stochastic input current, and investigate this choice of modeling in an event-based framework, making explicit very good performances. This method seems to be easily implementable in our present kernel, though this is out of the scope of the present work.

Conclusion

At a practical level, event-based simulation of spiking networks has been made available, using the simplest possible programmatic interface, as detailed previously. The kernel usage has been carefully studied, following the analysis proposed in [4] and detailed in Table 3

The present implementation thus offers a complementary alternative with respect to existing methods, and allows us to enrich the present spiking network simulation capabilities.

A Appendix: About gIF model normalization

Let us review how to derive an equation of the form of (2). We follow [20, 3, 38] in this section. We consider here a voltage dynamics of the form:

$$\frac{dV}{dt} + I^{leak} + I^{syn} + I^{gap} = I^{ext} + I^{adp} + I^{ion}$$

thus with leak, synaptic, gap-junction, external currents discussed in this section.

Membrane voltage range and passive properties

The membrane potential, outside spiking events, verifies:

$$V(t) \in [V_{\text{reset}}, V_{\text{threshold}}]$$

with typically $V_{\text{reset}} \simeq E_L \simeq -80mV$ and a threshold value $V_{\text{threshold}} \simeq -50mV \pm 10mV$. When the threshold is reached an action potential of about 1-2 ms is issued and followed by refractory period of 2-4 ms (more precisely, an absolute refractory period of 1-2 ms without any possibility of another spike occurrence followed by a relative refraction to other firing). Voltage peaks are at about $40mV$ and voltage undershoots about $-90mV$. The threshold is in fact not sharply defined.

The reset value is typically fixed, whereas the firing threshold is inversely related to the rate of rise of the action potential upstroke [2]. Here it is taken as constant. This adaptive threshold diverging mechanism can be represented by a non-linear ionic current [20, 3], as discussed in section 3.1.

From now on, we renormalize each voltage between $[0, 1]$ writing:

$$v = \frac{V - V_{\text{reset}}}{V_{\text{threshold}} - V_{\text{reset}}} \tag{14}$$

The membrane leak time constant $\tau_L \simeq 20ms$ is defined for a reversal potential $E_L \simeq -80\,mV$, as made explicit in (2).

The membrane capacity $C = S\,C_L \simeq 300pF$, where $C_L \simeq 1\,\mu F cm^{-2}$ and the membrane area $S \simeq 38.013\,\mu m^2$, is integrated in the membrane time constant $\tau_L = C_L/G_L$ where $G_L \simeq 0.0452\,mScm^{-2}$ is the membrane passive conductance.

From now on, we renormalize each current and conductance divided by the membrane capacity. Normalized conductance units are s^{-1} and normalized current units V/s.

Synaptic currents

In conductance based model the occurrence of a post synaptic potential on a synapse results in a change of the conductance of the neuron. Consequently, it generates a current of the form:

$$I^{syn}(V, \tilde{\omega}_t, t) = \sum_j G_j^+(t, \tilde{\omega}_t)\ [V(t) - E_+] + \sum_j G_j^-(t, \tilde{\omega}_t)\ [V(t) - E_-]\,,$$

for excitatory $+$ and inhibitory $-$ synapses, where conductances are positive and depend on previous spike-times $\tilde{\omega}_t$.

In the absence of spike, the synaptic conductance vanishes [21], and spikes are considered having an additive effect:

$$G_j^{\pm}(t, \tilde{\omega}_t) = \bar{G}^{\pm} \sum_n r^{\pm}(t - t_j^n)$$

while the conductance time-course $r^{\pm}(t-t_j^n)$ is usually modeled as an "exponential", "alpha" (see Fig. 10) or two-states kinetic (see Fig. 11) profile, where H is the Heaviside function (related to causality).

Note, that the conductances may depend on the *whole past history* of the network, via $\tilde{\omega}_t$.

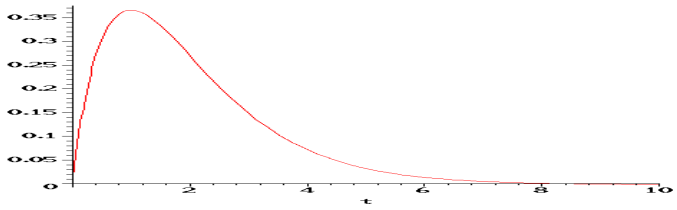

Figure 10: The "alpha" profile $\alpha(t) = H(t)\,\frac{t}{\tau}\,e^{-\frac{t}{\tau}}$ plotted here for $\tau = 1$. It is maximal at $t = \tau$ with $\alpha(\tau) = 1/e$, the slope at the origin is $1/\tau$ and its integral value $\int_0^{+\infty}\alpha(s)\,ds = \tau$ since $(\int\alpha)(t) = (\tau - t)\,e^{-\frac{t}{\tau}} + k$. This profile is concave for $t \in]0, 2\,\tau[$ and convex for $t \in]2\,\tau, +\infty[$, while $\alpha(2\,\tau) = 2/e^2$ at the inflexion point.

The "exponential" profile ($r(t) = H(t)\,e^{-\frac{t}{\tau}}$) introduces a potentially spurious discontinuity at the origin. The "beta" profile is closer than the "alpha" profile to what is obtained from a bio-chemical model of a synapse. However, it is not clear whether the introduction of this additional degree of freedom is significant here. Anyway, any of these can be used for simulation with the proposed method, since their properties correspond to what is stated in Fig. 1.

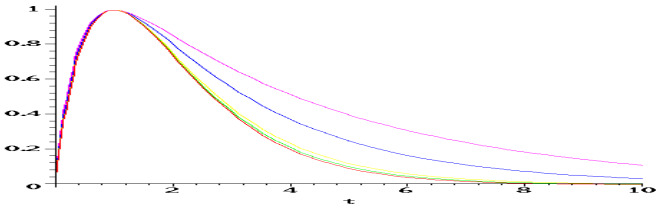

Figure 11: The two-states kinetic or "beta" profile $\beta(t) = H(t)\,\frac{1}{\kappa-1}\,(e^{-\frac{t}{\tau}} - e^{-\kappa\,\frac{t}{\tau}})$ is plotted with a normalized magnitude for the same $\tau = 1$ as the "alpha" profile but different $\kappa = 1.1, 1.5, 2, 5, 10$ showing the effect of this additional degree of freedom, while $\lim_{\kappa\to 1}\beta(t) = \alpha(t)$. The slope at the origin and the profile maximum can be adjusted independently with "beta" profiles. It is maximal at $t_\bullet = \tau\,ln(\kappa)/(\kappa-1)$ the slope at the origin is $1/\tau$ and its integral value τ/κ. The profile is concave for $t \in]0, 2\,t_\bullet[$ and convex for $t \in]2\,t_\bullet, +\infty[$.

There are typically, in real neural networks, 10^4 excitatory and about $2\,10^3$ inhibitory synapses. The corresponding reversal potential are $E_+ \simeq 0\,mV$ and $E_- \simeq -75\,mV$, usually related to AMPA and GABA receptors. In average: $\bar{G}_j^+ \simeq 0.66\,nS, \tau^+ \simeq 2\,ms$ and $\bar{G}_j^- \simeq 0.63\,nS, \tau^- \simeq 10\,ms$, for excitatory and inhibitory synapses respectively, thus about $570ms^{-1}$, $600ms^{-1}$ in normalized units, respectively. The coefficients $\bar{G}^{\pm}$ give a measure of the synaptic strength (unit of charge) and vary from one synapse to another and are also subject to adaptation.

This framework affords straightforward extensions involving synaptic plasticity (e.g. STDP, adjusting the synaptic strength), not discussed here.

Gap junctions

It has been recently shown, that many local inter-neuronal connections in the cortex are realized though electrical gap junctions [15], this being predominant between cells of the same sub-population [1]. At a functional level they seem to have an important influence on the synchronicity between the neuron spikes [23]. Such junctions are also important in the retina [50].

The electrotonic effect of both the sub-threshold and supra-threshold portion of the membrane potential $V_j(t)$ of the pre-junction neuron of index j seems an important component of the electrical coupling. This writes [50, 23]:

$$I^{gap}(V,t) = \sum_j G_j^* \left[[V_j(t) - V(t)] + E_\bullet \sum_n r(t - t_j^n) \right]$$

where G_j^* is the electrical coupling conductance, the term $V_j(t) - V(t)$ accounts for the sub-threshold electrical influence while and $E_\bullet$ parametrizes the spike supra-threshold voltage influence.

Regarding the supra-threshold influence, a value $E_\bullet \simeq 80mV$ corresponds to the usual spike voltage magnitude of the spiking threshold, while $\tau_\bullet \simeq 1ms$ corresponds to the spike raise time. Here $r()$ profile accounts for the action potential itself, slightly filtered by the biological media, while the gap junction intrinsic delay is of about $10\mu s$. These choices seem reasonable with respect to biological data [15, 23]

A step further, we propose to neglect the sub-threshold term for three main reasons. On one hand, obviously the supra-threshold mechanism has a higher magnitude than the sub-threshold mechanism, since related to action potentials. Furthermore, because of the media diffusion, slower mechanisms are smoothed by the diffusion, whereas faster mechanisms better propagate. On the other hand, this electrical influence remains local (quadratic decrease with the distance) and is predominant between cells of the same sub-population which are either synchronized or have a similar behavior, as a consequence $|V_j(t) - V(t)|$ remains small for cells with non-negligible electrical coupling. Furthermore, a careful analysis of such electrical coupling [23] clearly shows that the sub-threshold part of the contribution has not an antagonist effect on the neuron synchrony, i.e., it could be omitted without changing qualitatively the gap junction function.

As a conclusion, we are able to take gap junction into account with a minimal increase of complexity, since we obtain a form similar to synaptic currents, using very different parameters

External currents

Direct input (or external) current is often related to electro-physiological clamps. At another level of representation, the average activity of the neuron can be modeled by a constant or random input current. In both cases, the way the proposed simulation methods is proposed requires to assume such external current to be taken as constant between two spikes, i.e. having temporal variations small enough to be neglected. If not, it is easy to associate to the external current an event unit which fires at each new current value, in order the neuron to take this new value into account.

References

[1] Y. Amitai, J. Gibson, M. Beirleiner, S.L. Patrick, A.M. Ho, B.W.Connors, and D. Golomb. The spatial dimensions of electrically coupled networks of interneurons in neocortex. *J. Neurosci.*, 22:4142–4152, 2002.

[2] R. Azouz and C.M. Gray. Dynamic spike threshold reveals a mechanism for synaptic coincidence detection in cortical. In *Proceedings of the National Academy of Science*, volume 97, pages 8110–8115, 2000.

[3] R. Brette and W. Gerstner. Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of Neurophysiology*, 94:3637–3642, 2005.

[4] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris Jr., M. Zirpe, T. Natschläger, D. Pecevski, B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Vieville, E. Muller, A. P. Davison, S. El Boustani, and A. Destexhe. Simulation of networks of spiking neurons: a review of tools and strategies. *Journal of Computational Neuroscience*, 23(3):349–398, 2007.

[5] Romain Brette. Exact simulation of integrate-and-fire models with synaptic conductances. *Neural Computation*, 18(8):2004–2027, 2006.

[6] Romain Brette. Exact simulation of integrate-and-fire models with exponential currents. *Neural Computation*, 19(10):2604–2609, 2007.

[7] Giancarlo La Camera, Michele Giugliano, Walter Senn, and Stefano Fusi. The response of cortical neurons to in vivo-like input current: theory and experiment. *Biological Cybernetics*, 99(4-5):279–301, 2008.

[8] Giancarlo La Camera, Michele Giugliano, Walter Senn, and Stefano Fusi. The response of cortical neurons to in vivo-like input current: theory and experiment: Ii. time-varying and spatially distributed inputs. *Biological Cybernetics*, 99(4-5):303–318, 2008.

[9] B. Cessac. A discrete time neural network model with spiking neurons. rigorous results on the spontaneous dynamics. *J. Math. Biol.*, 56(3):311–345, 2008.

[10] B. Cessac, H.Rostro-Gonzalez, J.C. Vasquez, and T. Viéville. To which extend is the "neural code" a metric ? In *Neurocomp 2008*, 2008.

[11] B. Cessac and T. Viéville. On dynamics of integrate-and-fire neural networks with adaptive conductances. *Frontiers in neuroscience*, 2(2), jul 2008.

[12] C. G. Connolly, I. Marian, and R. G. Reilly. *Approaches to efficient simulation with spiking neural networks*, volume 15, pages 231–240. World Scientific, London, 2004.

[13] A. Destexhe, M. Rudolph, and D. Paré. The high-conductance state of neocortical neurons in vivo. *Nature Reviews Neuroscience*, 4:739–751, 2003.

[14] Alain Destexhe. Conductance-based integrate and fire models. *Neural Computation*, 9:503–514, 1997.

[15] M. Galarreta and S. Hestrin. Electrical synapses between gaba-releasing interneurons. *Nature Reviews Neuroscience*, 2:425–433, 2001.

[16] W. Gerstner and W. Kistler. *Spiking Neuron Models*. Cambridge University Press, 2002.

[17] M.L. Hines and N.T. Carnevale. Discrete event simulation in the neuron environment. *Neurocomputing*, pages 1117–1122, 2004.

[18] A.L. Hodgkin and A.F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–544, 1952.

[19] E. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, 2003.

[20] E.M. Izhikevich. Which model to use for cortical spiking neurons? *IEEE Trans Neural Netw*, 15(5):1063–1070, September 2004.

[21] C. Koch. *Biophysics of Computation: Information Processing in Single Neurons*. Oxford University Press: New York., 1999.

[22] Geehyuk Lee and Nabil Farhat. The double queue method: a numerical method for integrate-and-fire neuron networks. *Neural Networks*, 14(6-7):921–932, 2001.

[23] T. J. Lewis and J. Rinzel. Dynamics of spiking neurons connected by both inhibitory and electrical coupling. *Journal of Computational Neuroscience*, 14(3):283–309, 2003.

[24] W. Maass. Fast sigmoidal networks via spiking neurons. *Neural Computation*, 9:279–304, 1997.

[25] W. Maass and T. Natschlager. Networks of spiking neurons can emulate arbitrary hopfield nets in temporal coding. *Neural Systems*, 8(4):355–372, 1997.

[26] Wolfgang Maass and Christopher M. Bishop, editors. *Pulsed Neural Networks*. MIT Press, 2003.

[27] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann. Regulation of synaptic efficacy by coincidence of postsynaptic ap and epsp. *Science*, 275(213), 1997.

[28] Maurizio Mattia and Paolo Del Giudice. Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. *Neural Computation*, 12:2305–2329, 2000.

[29] A. Morrison, C. Mehring, T. Geisel, A.D. Aerstsen, and M. Diesmann. Advancing the boundaries of high connectivity network with distributed computing. *Neural Comput*, 17(8):1776–1801, 2005.

[30] A. Morrison, S. Straube, H.E. Plesser, and M. Diesmann. Exact subthreshold integration with continuous spike times in discrete-time neural network simulations. *Neural Computation*, 19:47–79, 2007.

[31] Anthony Mouraud, Hélène Paugam-Moisy, and Didier Puzenat. A distributed and multithreaded neural event-driven simulation framework. In *Int. Conf. on Parallel and Distributed Computing and Networks*, pages 393–398. ACTA Press, 2006.

[32] Jean-Pascal Pfister and W. Gerstner. Triplets of spikes in a model of spike timing-dependent plasticity. *J. Neurosci.*, 26:9673–9682, 2006.

[33] Alexander Rauch, Giancarlo La Camera, Hans-Rudolf Luscher, Walter Senn, and Stefano Fusi. Neocortical pyramidal cells respond as integrate-and-fire neurons to in vivo-like input currents. *J Neurophysiol*, 90(3):1598–1612, 2003.

[34] J. Reutimann, M. Guigliano, and S. Fusi. Event-driven simulation of spiking neurons with stochastic dynamics. *Neural Comput.*, 15:811830, 2003.

[35] O. Rochel. *Une approche événementielle pour la modélisation et la simulation de neurones impulsionnels*. PhD thesis, Université Henri Poincaré - Nancy 1, 2004.

[36] O. Rochel and D. Martinez. An event-driven framework for the simulation of networks of spiking neurons. In *Proc. 11th European Symposium on Artificial Neural Networks*, pages 295–300, 2003.

[37] S. Rotter and M. Diesmann. Exact digital simulation of time-invariant linear systems with application to neuronal modeling. *Biol. Cybern.*, 81:381–402, 1999.

[38] M. Rudolph and A. Destexhe. Analytical integrate and fire neuron models with conductance-based dynamics for event driven simulation strategies. *Neural Computation*, 18:2146–2210, 2006.

[39] M. Rudolph and A. Destexhe. How much can we trust neural simulation strategies? *Neurocomputing*, 2007. To appear.

[40] Berthold Ruf. *Computing and Learning with Spiking Neurons - Theory and Simulations*. PhD thesis, Institute for Theoretical Computer Science, Univ. Graz, Austria, 1998.

[41] Benjamin Schrauwen. *Towards Applicable Spiking Neural Networks*. PhD thesis, Universiteit Gent, Belgium, 2007.

[42] Jonathan Touboul. Bifurcation analysis of a general class of non-linear integrate and fire neurons. Research Report RR-6161, Odyssee, INRIA, 2007.

[43] Jonathan Touboul. Bifurcation analysis of a general class of nonlinear integrate-and-fire neurons. *SIAM Journal on Applied Mathematics*, 68(4):1045–1079, 2008.

[44] J.D. Victor. Spike train metrics. *Current Opinion in Neurobiology*, 15(5):585–592, 2005.

[45] T. Viéville, S. Chemla, and P. Kornprobst. How do high-level specifications of the brain relate to variational approaches? *Journal of Physiology - Paris*, 101(1-3):118–135, 2007.

[46] T. Viéville and O. Rochel. One step towards an abstract view of computation in spiking neural-networks. In *International Conf. on Cognitive and Neural Systems*, 2006.

[47] Tim P. Vogels and L. F. Abbott. Signal propagation and logic gating in networks of integrate-and-fire neurons. *J. Neuroscience*, 25:1078610795, 2005.

[48] Tim P. Vogels, Kanaka Rajan, and L. F. Abbott. Neural network dynamics. *Annu. Rev. Neurosci.*, 28:35776, 2005.

[49] C. J. Wilson, A. Weyrick, D. Terman amd N. E. Hallworth, and M. D. Bevan. A model of reverse spike frequency adaptation and repetitive firing of subthalamic nucleus neurons. *J. Neurophysiol*, 91:1963–1980, 2004.

[50] A. Wohrer and P. Kornprobst. Virtual retina : A biological retina model and simulator, with contrast gain control. *Journal of Computational Neuroscience*, 2008. DOI 10.1007/s10827-008-0108-4.

[51] G. Zheng, A. Tonnelier, and D. Martinez. Voltage-stepping schemes for the simulation of spiking neural networks. *J. Comput Neurosci.*, 26, 2008.

Acknowledgment: Partially supported by the ANR MAPS & the MACCAC ARC projects. We are especially thankful to Hélène Paugam-Moisy for precious advices and inputs regarding this piece of work.

Features		
Clock-based	: can it simulate clock-based strategies ?	yes
	: in this case, does it use extrapolation for spike times ?	useless[1]
Event-based	: can it simulate event-based strategies ?	yes
	: in this case, is the integration scheme exact ?	yes[2]
Parallelism	: does it support parallel processing ?	no[7]
Graphics	: does it have a graphical interface ?	no, but a programmatic interface
Simple analysis	: is it possible to perform simple analysis ?	yes with visualization
Complex analysis	: can more complex analysis be done ?	it can[3]
Interface	: is interface to outside signals possible ?	indeed[4]
	: is it interoperable with other simulators ?	yes[4]
Save option	: can simulation be halted / resumed ?	yes
Neuron models	: can it simulate HH models ?	it can[5]
	: can it simulate leaky IF models ?	yes
	: can it simulate multivariate IF models ?	it can[5]
	: can it simulate conductance-based synaptic interactions ?	yes
	: can it simulate short-term plasticity ?	it can[5]
	: can it simulate long-term plasticity ?	it can[5]
	: can it simulate compartmental models with dendrites ?	no
Usage		
Development	: is it currently developed ?	yes, still α-version
	: how may developers yet ?	half-time researcher + students
Support	: is it supported	yes
	: what kind of support	email + phone
	: are they user cooperative tools ?	not yet, tools available[6]
Manual	: are there tutorials and reference material available ?	yes
	: are there published books on the simulator ?	no (useless)
	: is there a list of publications of articles that used it ?	yes
Import/export	: is standard (XML) specification import/export available ?	it can[3]
Web site	: is there a web site where all can be found ?	`http://enas.gforge.inria.fr`
Source code	: are there codes available on the web ?	yes
Operating system	: does it run under Linux	yes, tested
	: does it run under Max-OS X	yes, tested
	: does it run under Windows	likely (untested)
Interoperability	: using which language can it be used ?	C/C++, Python, Java, Php
	: can it be used from other platforms ?	yes[8]

Notes:

1: Since clock/event-based mechanisms can be mixed in a event-based simulation, spike times extrapolation is no more to be used, but exact event-time instead.

2: Exact integration scheme is to be used when allowed by the model, lower-bound spike time evaluation is a new alternative proposed here when the former is not possible.

3: More complex analysis and XML specification import/export is indeed possible, using this kernel within the PyNN environment. The goal was to *only* develop here, what was not available elsewhere. The API has been carefully designed for this purpose.

4: The interface capability is a key feature of this middle-ware implementation, including in real time applications using spike computations.

5: Plasticity and other existing models, not discussed here, can be implemented with this middle-ware, and STDP is already considered at that time. The real goal is however not to implement "all" models, other simulators do that better, but to propose also alternatives to existing models, as discussed in this paper.

6: The present development is installed on a forge, thus has all forum/bug-tracking/user-resquest-ticketting, etc.. available.

7: Though parallelism is not available yet, and the issue not addressed here, the network simulation kernel has been designed and implemented in order to be able to connect to other kernels, via input/output events. It is thus a feasible extension to run several kernels in parallel, with the drawback that the slower kernel is going to drive other kernels local times.

8: Links with these external platforms such as PyNN (thus NEURON, MvaSpike, ..), NeuroConstruct are made available by the multi-language operability, while Scilab and Matlab usage is documented.

Table 3: Summary of the main features of the implemented event-based simulation kernel, using the criteria proposed to compare existing simulators, see text for details.

Centre de recherche INRIA Sophia Antipolis – Méditerranée
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399